IN THE CLAIMS

This listing of claims replaces all prior listings:

Claims 1-28. (Canceled).

29. (Currently Amended) A method in a data processing system for developing a data flow program comprising code segments distributed between blocks of memory, the method comprising the steps of:

generating a graph that represents the data flow program comprising code segments distributed between the blocks of memory, the graph comprising nodes corresponding to selected ones of the blocks and arcs corresponding to dependency relationships between the nodes; and

receiving an optimization command to manipulate the generated graph to improve performance of the data flow program.

30. (Previously Presented) The method of claim 29, further comprising the step of performing the optimization command.

31. (Previously Presented) The method of claim 30, further comprising the step of performing performance analysis on the data flow program in accordance with the optimization command.

32. (Previously Presented) A method according to claim 29, wherein the nodes are placed in an execution queue for processing, and wherein the optimization command specifies re-ordering the nodes in the execution queue.

33. (Previously Presented) A method according to claim 29, wherein the nodes are characterized by node execution times, and wherein the optimization command specifies setting one of the node execution times, and further comprising the step of simulating execution of the data flow program in accordance with the node execution times.

34. (Previously Presented) A method according to claim 29, wherein the blocks are assigned data operated on by the data flow program and wherein the optimization command specifies setting revised data for a selected block.

35. (Previously Presented) A method according to claim 29, wherein the optimization command specifies a performance comparison between selected nodes.

36. (Currently Amended) A method in a data processing system for developing a data flow program comprising code segments that operate on data in memory, the method comprising the steps of:

dividing a memory area into blocks and associating each block with at least a portion of the data in memory and with at least one code segment;

generating a graph representation of the data flow program, the graph representation comprising nodes associated with the blocks, and arcs associated with dependencies between the blocks; and

performing an optimization command to manipulate the generated graph to improve performance of the data flow program.

37. (Previously Presented) The method of claim 36, further comprising the step of performing performance analysis on the data flow program.

38. (Previously Presented) The method of claim 36, further comprising the step of entering the nodes in a queue for execution, and wherein the optimization command specifies reordering the nodes in the queue.

39. (Previously Presented) The method of claim 37, wherein the step of performing performance analysis comprises the step of determining execution time for the data flow program.

40. (Previously Presented) The method of claim 37, wherein the step of performing performance analysis comprises the step of simulating execution of the nodes in the graph.

41. (Previously Presented) The method of claim 37, wherein the nodes are characterized by node execution times, and wherein the optimization command specifies a reduced node execution time for one of the nodes, and wherein the step of performing performance analysis comprises the step of determining execution time for the data flow program in accordance with

the reduced node execution time.

42. (Previously Presented) The method of claim 37, wherein the optimization command specifies a memory bandwidth, and wherein the step of performing performance analysis comprises the step of determining execution time for the data flow program in accordance with the memory bandwidth.

43. (Previously Presented) The method of claim 36, wherein the optimization command specifies a modification to at least a portion of the data.

44. (Currently Amended) A computer-readable medium containing instructions that cause a data processing system to perform a method for developing a data flow program comprising code segments that operate on data in memory, the method comprising the steps of:

dividing the memory into blocks and associating each block with at least a portion of the data in memory and with at least one code segment;

generating a graph representation of the data flow program, the graph representation comprising nodes associated with the blocks, and arcs associated with dependencies between the blocks; and

performing an optimization command to manipulate the generated graph to improve performance of the data flow program.

45. (Previously Presented) The computer-readable medium of claim 44, further comprising the step of performing performance analysis on the data flow program.

46. (Previously Presented) The computer-readable medium of claim 44, further comprising the step of entering the nodes in a queue for execution, and wherein the optimization command specifies reordering the nodes in the queue.

47. (Previously Presented) The computer-readable medium of claim 45, wherein the step of performing performance analysis comprises the step of determining execution time for the data flow program.

48. (Previously Presented) The computer-readable medium of claim 45, wherein the step of performing performance analysis comprises the step of simulating execution of the nodes in the graph.

49. (Previously Presented) The computer-readable medium of claim 45, wherein the nodes are characterized by node execution times, and wherein the optimization command specifies a reduced node execution time for one of the nodes, and wherein the step of performing performance analysis comprises the step of determining execution time for the data flow program in accordance with the reduced node execution time.

50. (Previously Presented) The computer-readable medium of claim 44, wherein the optimization command specifies a modification to at least a portion of the data.

51. (Currently Amended) A data processing system comprising:

a memory comprising a data flow program and a data flow development tool that generates a graph representation of the data flow program by associating data in memory processed by the data flow program to blocks in the memory, by associating code segments of the data flow program to the blocks, and by determining dependencies between the blocks, that executes code segments in parallel using multiple threads, and that performs an optimization command to manipulate the generated graph to improve performance of the data flow program; and

a processor that runs the data flow development tool.

52. (Previously Presented) The data processing system of claim 51, further comprising a queue in memory in which the blocks are placed for execution, and wherein the optimization comprises reordering the blocks in the queue.

53. (Previously Presented) The data processing system of claim 51, wherein the optimization command comprises an execution time specification of for one of the blocks.

54. (Previously Presented) The data processing system of claim 51, wherein the optimization command comprises a block performance comparison command.

55. (Previously Presented) The data processing system of claim 51, wherein the optimization command comprises a block data modification command.

56. (Currently Amended) A data processing system for developing a data flow program comprising code segments that operate on data in memory, the data processing system comprising:

means for apportioning a memory into regions and associating the data in the memory and the code segments of the data flow program with the regions;

means for determining dependencies between the regions;

means for generating a graph representation of the data flow program, the representation comprising nodes associated with the regions, and dependencies between regions; and

means for obtaining an optimization command specifying an optimization to ~~on~~ the data flow program by manipulating the generated graph.